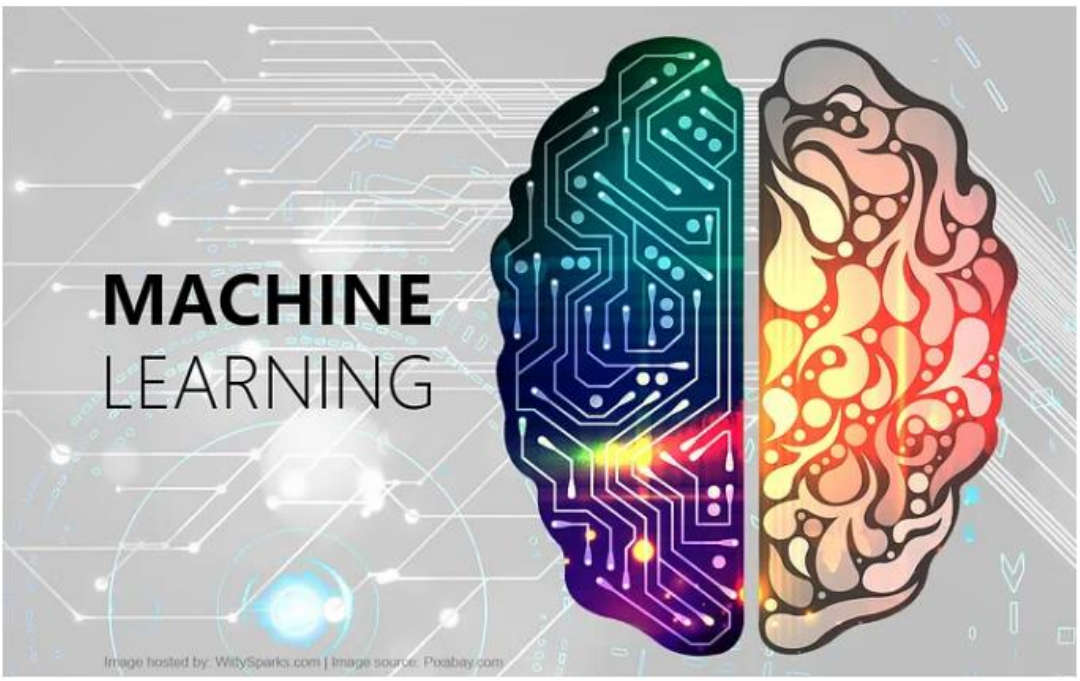
פרויקט למידת מכונה - חלק ב

מגיש: נועם גנים



שם מרצה: בעז לרנר

שם מתרגל: גלעד ארז

הכנת הנתונים לאימון ובחינה:

סט הנתונים שאנחנו עובדים עליו הוא הסט שסיימנו לעבוד בחלק א לאחר הורדת והשלמת עמודות ורשומות ריקות ולא הגיוניות.

השיקולים שלקחנו לקביעת גודל סט הנתונים הם:

אם נקצה הרבה נתונים לסט הבחינה לא יהיה לנו מספיק נתונים לאימון טוב של המודל.

אם נקצה קצת נתונים לבחינה הרעש בסט הבחינה יהיה גדול מדי.

לכן בחרנו כמות שבין 2 הגבולות האלה ובהתאם להמלצה, 900 רשומות לסט הבחינה שהם 10% מסך הנתונים הקיימים. בחרנו את אופן החלוקה 10-90 כי קראנו שעבור סט נתונים קטן מ 10 אלף רשומות נהוג לחלק ל 10%-90.

חילקנו את סט הנתונים לפי העמודה של satisfaction באופן ששומר על הפרופורציה בין כמות הערכים שמסווגות satisfied וכמות הערכים שמסווגות neutral or dissatisfied באמצעות שיטת Hold-Out.

עצי החלטה:

1. הכנת הנתונים להתאמה לעצי החלטה:

כדי להתאים את הנתונים לעצי החלטה המרנו את כל המשתנים הקטגוריאליים עם 2 שדות ל 1 או 0. כאשר למשתנה קטגוריאליים שלא ניתן לסידור היה יותר מ 2 שדות יצרנו עמודה לכל אחד מהשדות והוא קיבל ערכים של 1 אם הוא מתקיים או 0 אם הוא לא באותה רשומה.

את המשתנים הרציפים נרמלנו כדי שהערכים יהיו בין 0 ל 1 לפי הנוסחה של minmax.

2. עץ החלטה מלא:

הרצנו את המודל הדיפולטיבי, כאשר השתמשנו במדד **entropy**. אחוז הדיוק שהתקבל על סט **האימון** היה 100% – Overfitting, לעומת זאת, הדיוק שנמדד על סט **הבחינה** היה פחות מדויק-81.9%, לפיכך אנו מסיקים כי בשל התאמת יתר על סט האימון המודל למד את הנתונים בצורה מאוד מדויקת, ולכן הוא "זוכר" את הרעש מהנתונים במהלך האימון ולא מצליח להכליל טוב על סט חדש (סט הבחינה) עם רעש שונה שהוא לא ראה.

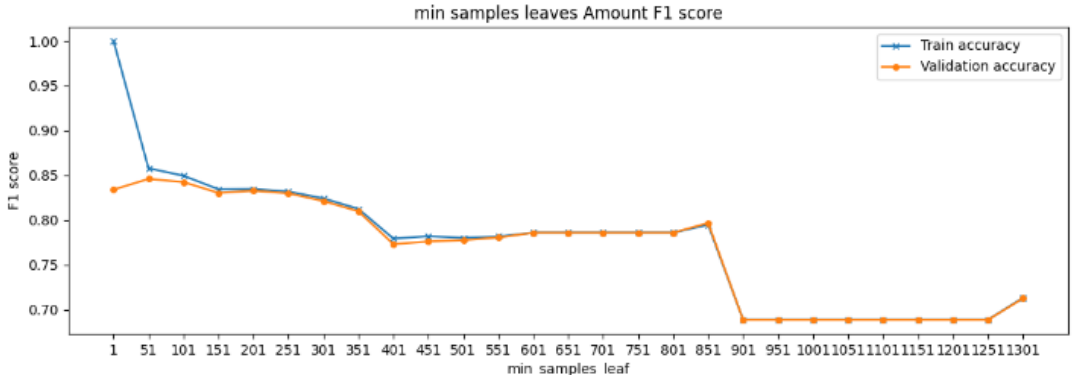
עץ מלא יביא לתוצאה של 100% דיוק על סט האימון ברוב המקרים, במיוחד כאשר המודל מורכב ומעמיק.

עץ מלא לא יביא לדיוק של 100% כאשר נתוני האימון סותרים, כלומר רשומות זהות עם משתנה מטרה שונים, עץ ההחלטות לא יוכל להפריד ביניהן בצורה מושלמת ולכן לא יגיע ל-100% דיוק.

3. כוונון פרמטרים:

על מנת להימנע מ Overfitting, נתמודד עם הרעש בדאטה ע"י קיטום העץ.

בבדיקה ראשונית למשתנה Min_Sample_Leaves ע"פ התרשים מטה, אנו רואים כי החיזויים מחושבים באחוזי דיוק טובים החל מעץ הכולל 51 עלים, ועד לעץ הכולל 251 עלים. מתחת ל50 עלים נימצא במצב של underfitting, ומעל 251, נימצא במצב המתקרב לoverfitting נבחן את הטווח בין 51-251 עלה בהמשך עם כלל הפרמטרים.



4. גיזום העץ:

בחרנו לגזם את העץ לפי שיטת: **Post-Pruning.**

בתהליך הגיזום נרצה להסתכל על פרמטר ה-Class weight , עבור שני מקרים:

כאשר ערכו None (נותן משקל שווה לשני הקלאסים) וכאשר ערכו balanced

(מאזן משקולות בהתאם למספר הדגימות בכל קלאס).

בנוסף, נרצה להסתכל על פרמטר criterion, עבור שני מדדים: מדד Gini , מדד Entropy

על מנת לכוונן את ההיפר פרמטרים האלה - השתמשנו ב- grid search.

תוצאות החיפוש :

std test score	Mean test score	Mean train score	Max depth	Criterion	Class weight	Min samples leaf
0.0127	0.8533	0.8543	19	entropy	balanced	85
0.0127	0.8533	0.8543	18	entropy	balanced	85
0.0127	0.8533	0.8543	17	entropy	balanced	85
0.0127	0.8533	0.8543	15	entropy	balanced	85
0.0127	0.8533	0.8543	14	entropy	balanced	85
0.0127	0.8533	0.8543	16	entropy	balanced	85
0.0127	0.8533	0.8543	13	entropy	balanced	85
0.0127	0.8533	0.8543	10	entropy	balanced	85
0.0127	0.8533	0.8543	11	entropy	balanced	85
0.0127	0.8533	0.8543	8	entropy	balanced	85

מתוך תוצאות הgrid search נבחרו הפרמטרים הבאים:

<u>עומק מקסימלי (max depth)</u>- 8, המשמעות היא שנבחן טווח של בין 1-8 התפתחויות בעץ. אם נגדיל ערך פרמטר זה נחזור לבעיה של (Over fitting) מעבר לזה העץ מתחיל לשנן את הנתונים ואת הרעש שלהם ויתחיל לענות בצורה שמסתמכת יותר מידי על סט הנתונים שיש לנו במקום לחזות בצורה כללית וחכמה.

<u>min sample leaf</u>- 85 עלים.

<u>class weight</u>- balanced.

<u>criterion</u> – entropy, הפונקציה למדידת איכות הפיצול.

המודל הנבחר:

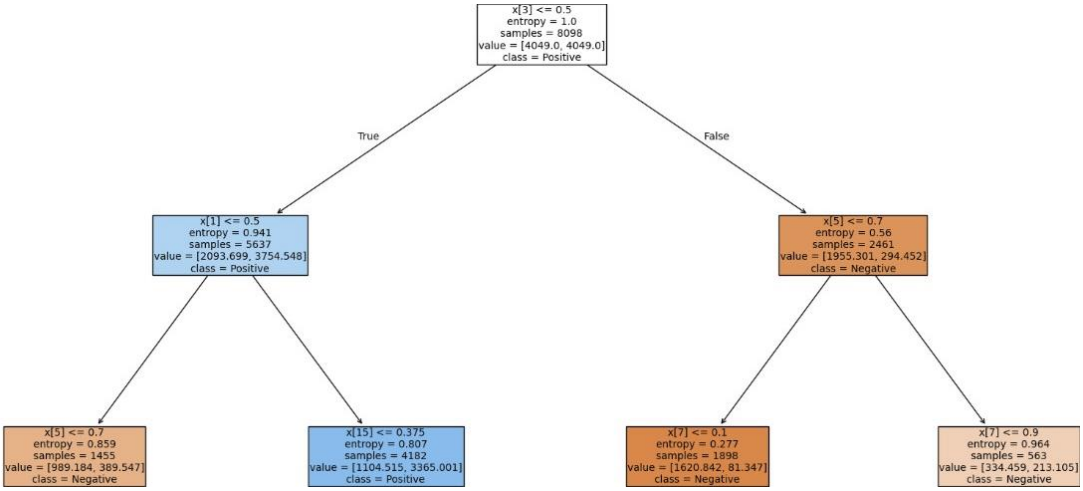
בעל 85.19% דיוק על סט האימון ו- 87.19% דיוק על סט הוולידציה באמצעות grid search.

```
F1 Training score: 0.8519
F1 Testiing score: 0.8719
```

עץ חלקי הינו פשוט יותר, מכיל פחות צמתים ועלים, ומתמקד במאפיינים העיקריים בנתונים. כתוצאה מכך, הוא פחות מדויק על סט האימון, אך עשוי להיות אמין יותר בהתמודדות עם נתונים לא מוכרים.

חלק מהעץ שהתקבל :

שקיפות זו עוזרת להבין את ההיגיון מאחורי החלטות המודל ומאפשרת לזהות אילו תכונות הן המשפיעות ביותר בקביעת התוצאה, כך ניתן לראות אילו פיצ'רים הם בעלי חשיבות גבוהה יותר.



ניתן לראות את החשיבות של המשתנים בטבלה מטה, כך שהמשתנה X3 הינו (Type of travel) בעל החשיבות הגבוהה ביותר. לכן הוא נמצא בשורש העץ. ניתן לראות גם את המשתנים X1 (customer type) X5, (inflight Wi-Fi service) ו- X7 (Ease of online booking). החשיבות של משתנים אלה הינה לפי הסדר, כלומר כאשר מדובר במשתנה ששייך לרמה יותר נמוכה בעץ אז הוא בעל חשיבות יותר נמוכה, ניתן לוודא זה בטבלת חשיבות המשתנים.

ניתן לראות כי חשיבות הערכים של הפיצ'רים מטה שערכם 0, מראה כי הם כנראה לא חשובים למשימת סיווג זו או שעץ החלטות לא יכול להעזר בהם לשיפור סיווג ואולי אלגוריתם אחר יצליח יותר.

בחלק א' חשדנו בכמה מהמשתנים(צבעי מטוס – 1,2,3) שכנראה מיותרים, בחלק זה הדבר מקבל אישוש לגבי חשיבותם.

Feature_name	Importance
Gender	0.0
Customer Type	0.1502
Age	0.0057
Type of Travel	0.281
Flight Distance	0.01
Inflight wifi service	0.1351
Departure/Arrival time convenient	0.0
Ease of Online booking	0.0572
Gate location	0.0
Food and drink	0.0
Seat comfort	0.0213
On-board service	0.0
Baggage handling	0.016
Checkin service	0.0165
Inflight service	0.0085
Cleanliness	0.1844
Departure Delay in Minutes	0.0023
Service quality	0.1038
Business	0.0056
Eco	0.0
Eco Plus	0.0
1	0.0024
2	0.0
3	0.0

הספירה של Xים מתחילה מאפס (לדוגמה Gender-0 , Customer type-1 וכו').

רשתות נוירונים:

1. הכנת הנתונים להתאמה לרשתות נוירונים:

המשכנו עם הנתונים מהסעיף של עצי ההחלטה בטבלה df1.

שכפלנו את הנתונים לטבלה חדשה בשם df2 .

יצרנו שוני בין הטבלאות על מנת לבדוק איזה נרמול עדיף לתוצאות טובות יותר.

בטבלה df1 נרמלנו את המשתנים הרציפים בעזרת minmax_scaling() בסקלה בין 0 ל 1.

בטבלה df2 נרמלנו את המשתנים הרציפים בעזרת StandardScaler() כך שהמשתנים יהיו בטווח נורמלי סטנדרטי.

2. רשת נוירונים דיפולטיבית

בחלק זה הרצנו את ערכי ברירת המחדל של המודל. הסבר על הקונפיגורציה:

<u>נוירונים בשכבת הכניסה:</u> מספר הנוירונים בשכבת הכניסה נקבע לפי מספר המשתנים כקלט במודל (X2 עבור df2 ו-X עבור df1).

<u>מספר שכבות חבויות ומספר נוירונים:</u> שכבה אחת חבויה של נוירונים שמכילה 100 נוירונים.

<u>פונקציית אקטיבציה:</u> ברירת המחדל היא 'relu' , שמחזירה ערך f(x) = max(0, x).

<u>נוירונים בשכבה האחרונה:</u> זו שכבת הפלט, במקרה שלנו (משימת קלסיפיקציה) הפלט בינארי ולכן יש נוירון יחיד שייצג את הסיכוי של סמפל מסוים להיות משויך לקלאס 1: Positive או 0: Negative

קיבלנו כי המודל לא מתכנס לאחר 200 איטרציות שהם ערכי ברירת המחדל של max_iter, כלומר עדיין לא הגיע לסיום משימת הלמידה ולתנאי העצירה, בהמשך נרצה להוסיף איטרציות כדי להגיע להתכנסות .

מדד הביצוע שבחרנו לבדוק הוא f1, התוצאות שקיבלנו על סט האימון והמבחן:

```
f1 Training score: 0.9381
f1 Testiing score: 0.8742
```

```
f1 Training score2: 0.9381
f1 Testiing score2: 0.8742
```

קיבלנו כי אין הבדל בתוצאות מדד של f1 על סט האימון וסט המבחן בין df1 ל df2 עם המודל הדיפולטיבי, ולכן נבחר שרירותית להמשיך עם df2. ניתן לראות כי הדיוק של סט המבחן 0.874 ושל סט האימון 0.9381

3. כיוונון פרמטרים - השפעה לכל היפרפרמטר שנבחר בתהליך

<u>Hidden Layer Size - גודל שכבה נסתרת</u>

מבנה הרשת כולל מספר וגודלן של השכבות הנסתרות, השכבה הראשונה בעלת משמעות גבוהה שמעבדת את הקלט ברמה כללית ביחס לשכבה אחריה ולפי כך נוצר ייצוג היררכי עולה ברמת הניתוח למציאת דפוסים ומשתנים מורכבים משכבה לשכבה ולפיכך רמת ה- abstraction עולה ומונעת התאמת יתר, לכן מספר שכבות קטן מידי עשוי להוביל למודל פשוט עם יכולת הכללה פחות טובה ומספר שכבות גדול מידי יכול להוביל להתאמת יתר. אחת ההמלצות של מדעני נתונים בחברת גוגל להשתמש בין 3-5 שכבות לסט מורכב.

מכיוון שיש לנו 21 משתנים וסט הנתונים לא מתפלג ליניארית -לכן לא נבחר נוירון בודד בשכבה, לפי מה שמצאנו באינטרנט ובמחקרים הומלץ שבכל שכבה לא יהיו יותר נוירונים מהכפלה של הכמות שבשכבת

הקלט והפלט שבמקרה שלנו 21 לפי מספר המשתנים ושכבת הפלט =1 , לפיכך אחרי כמה ניסוים אנו בוחנים טווח 2 עד 50 לכל שכבה.

Batch size - גודל אצווה

גודל האצווה קובע את מספר הדגימות המעובדות לפני עדכון פרמטרי המודל.
גדלי אצווה קטנים יותר מספקים בדרך כלל אומדן מדויק יותר של השיפוע אך עלולים לגרום לעדכונים רועשים יותר, שעלולים להוביל לזמני אימון ארוכים יותר. גדלי אצווה גדולים יותר מציעים עדכונים חלקים יותר אך עשויים לדרוש יותר זיכרון ועלולים להוביל להתכנסות לא אופטימלית. אחרי חיפוש באינטרנט ראינו שההמלצה לגודל קבוצה אופטימלי הוא חזקה של 2, והערכים הנפוצים הם 32, 64, 128, 256 . לפיכך נבחן את האצוות בסדרי גודל אלו.

Max iter - מקסימום איטרציות

פרמטר זה קובע את המספר המרבי של איטרציות לאימון. הגדלת מספר האיטרציות מאפשרת למודל יותר הזדמנויות ללמוד, מה שיכול לשפר את הביצועים. עם זאת, יותר מדי איטרציות עלולות להוביל להתאמת יתר. בחרנו כמות איטרציות של 100 מכיוון שכמות גדולה מזה לקחה המון זמן להרצה.

Learning rate init

קצב הלמידה שולט בגודל הצעד לעדכון פרמטרי מודל במהלך האימון.
קצב למידה גבוה יותר יכול להאיץ את האימונים, אך מסתכן בחריגה של ערכי פרמטרים אופטימליים, מה שמוביל להתכנסות לקויה. שיעור למידה נמוך יותר מציע עדכונים מדויקים יותר אך יכול לגרום לזמני אימון ארוכים. במודל ברירת מחדל קצב הלמידה שווה 0.001 והוא קבוע לאורך כל הלמידה, מכיוון והמודל שלנו לא התכנס במודל הדיפולטיבי יתכן שקצב הלמידה שלנו יחסית נמוך והמודל נתקע במינימום לוקאלי בהשוואה לגודל האצווה שלנו, נבחן טווח ערכים יחסית רחב כדי לאפשר ללמידה להתמודד עם גדלי אצווה שבחרנו לבחון. הטווח שנבחן נע בין: 0.0005, 0.001, 0.005, 0.01, 0.05.

Learning rate

בוחן את קצב הלימדה בקונפיגורציה של constant- קצב הלמידה קבוע לאורך הלמידה ו- adaptive - כאשר פונקציית ההפסד לא קטנה יותר מ- tolerance קצב הלמידה יחולק ב 5 ויאפשר התכנסות לערך המינימלי וזה גם יאפשר להתמודד עם גודל אצווה יותר קטן.

Activation – פונקצית אקטיבציה

פונקציית הפעלה עבור השכבות הנסתרות. לפונקציות הפעלה שונות יש מאפיינים שונים.
אנחנו החלטנו לבחור לבחון את relu, logistic(sigmoid), tanh כי ההמלצה היא להשתמש באחת מהן כאשר המטרה שלנו היא קלסיפיקציה בינארית, רק עבור ה-output (שכבת היציאה, האחרונה) פונקציית ההפעלה נבחרת אוטומטית לפי סוג המשימה, למשל עבור קלסיפיקציה בינארית החבילה תשתמש ב logistic.

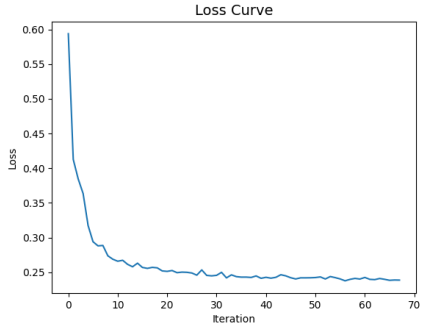
Solver

אלגוריתם האופטימיזציה משמש לאימון המודל בכך שמעדכן את המשקולות.

אנחנו נבחן שתי פונקציות Adam ו- sdg כי שתי הגישות מבוססות על gradient descent עם אדפטציות שונות. Sgd היא פשוטה ויעילה אך יכולה להיות איטית ורגישה לקצב הלמידה. adam מתאים את קצב הלמידה עבור כל פרמטר, מה שגורם לרוב להתכנסות מהירה וחזקה יותר.

4. המודל האופטימלי שנבחר:

std test score	Mean test score	Mean train score	Hidden Layer Size	Batch Size	Learning Rate Init	Learning Rate	Activation	Solver
0.0107	0.8856	0.8935	(7, 12, 47, 2, 22)	32	0.01	adaptive	logistic	adam



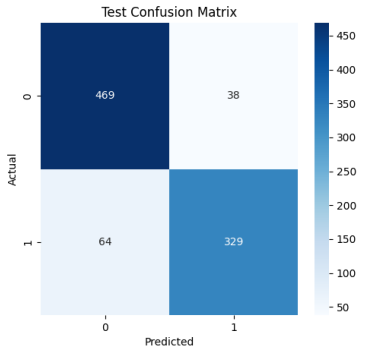
רמת הדיוק f1 לפי המודל האופטימלי על סט האימון היא 0.8935 ועל סט הוולידציה 0.8856 . הרמות כמעט זהות למודל הדיפולטיבי, כנראה בגלל שלחלק מהמשתנים שלנו יש יכולת גבוהה להבדיל בין הקלאסים. מנגד, יכולת הכללה של המודל טיפה יותר טובה כיון שמתכנסת סביב 60 איטראציות עם שיפוע יותר חד סביב האיטראציות הראשונות אפשר לשייך את זה לקצב הלימוד שבחר המודל 0.01.

ניתן לראות לפי 10 המודלים הכי טובים שקבלנו שההפרש בין הביצועים שלהם כמעט זהה מבחינת אחוזי דיוק עם קומבינציות שונות של hyperparameters זו אינדיקציה שהפיצ'רים שלנו הם טובים כך שהמודל הצליח לבצע קלסיפיקציה עבור הקומבינציות הללו.

Random search results

std test score	Mean test score	Mean train score	Hidden Layer Size	Batch Size	Learning Rate Init	Learning Rate	Activation	Solver
0.0107	0.8856	0.8935	(7, 12, 47, 2, 22)	32	0.01	adaptive	logistic	adam
0.0089	0.883	0.8978	(7, 27, 32, 22, 17)	32	0.01	adaptive	relu	adam
0.0072	0.8829	0.9129	(37, 47, 22, 42)	256	0.005	adaptive	tanh	sgd
0.0096	0.8821	0.8923	(42, 17, 32, 47, 27)	64	0.001	constant	logistic	adam
0.0061	0.8814	0.8952	(7, 32, 17, 22, 42)	64	0.01	constant	logistic	adam
0.0101	0.8802	0.8999	(32, 7, 47, 27, 42)	32	0.001	constant	logistic	adam
0.0109	0.8787	0.8819	(2, 12, 32, 27, 22)	64	0.01	adaptive	relu	sgd
0.008	0.8785	0.8871	(7, 2, 37, 32, 22)	256	0.05	constant	relu	adam
0.0124	0.8785	0.9203	(32, 2, 42, 37)	256	0.001	constant	relu	adam

מטריצת מבוכה




```
Test Confusion Matrix:
 [[469  38]
 [ 64 329]]
Test Set: TP=329, TN=469, FP=38, FN=64
```

מטריצת המבוכה מחושבת על סט הוולידציה. ניתן לראות שה-True Positive הוא 329, כלומר מתוך הסאמפלים שהקלאס שלהם הוא חיובי הצלחנו לסווג כ-83.74% מהסאמפלים נכון. ה-True Negative הוא 469 כלומר הצלחנו לסווג נכון כ-92.28% מהסאמפלים השליליים נכון. בהתבוננות בתוצאות ניתן לראות שהמודל מצליח לחזות בדיוק די גבוה.

אנו מסיקים שהמודל שלנו רגיש למשימת הקלסיפיקציה, אך מעט פחות רגיש מהמודל הדיפולטיבי.

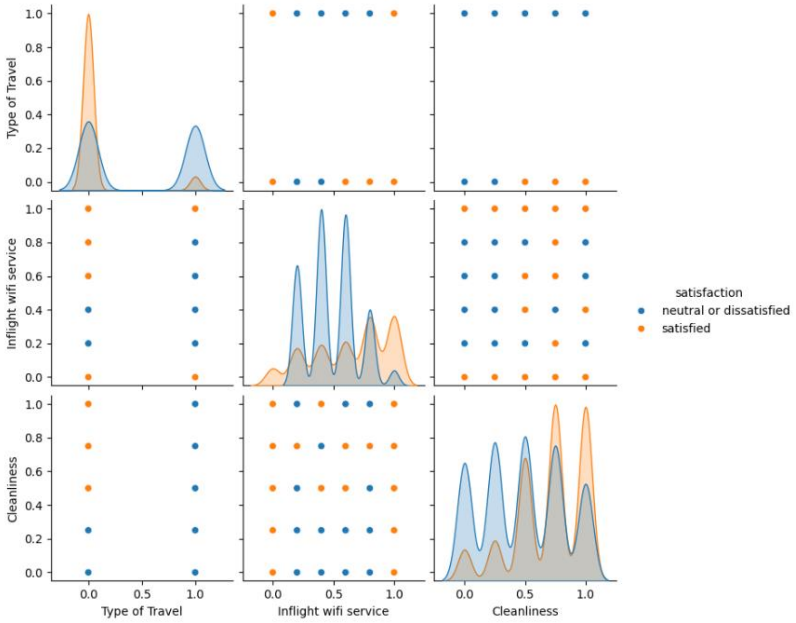
Clustering -אשכול:

אשכול הוא פעולה תחת למידה לא מונחית, האלגוריתם לומד בעצמו איך לקבץ את הדוגמאות לקטגוריות משמעותיות ,בלי צורך להגיד לו מראש מה הלייבלים.

כך שבכל אשכול יתקבצו סמפלים בעלי דמיון על פי הפיצ'רים שלהם.

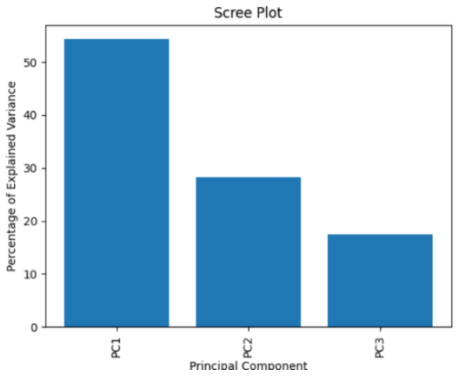
על ידי תהליך הלמידה הקודם הצלחנו לדעת איזה פיצ'רים יותר תורמים לתהליך הלמידה, ולכן החלטנו בתהליך של האשכול לתת לאלגוריתם מספר קטן של משתנים שמספרים את המוספר באופן יותר חזק מהשאר. רצינו לראות מבט על לנתונים לפי משתנים אלה ולראות עד כמה הם מצליחים להפריד בין שתי הלייבלים שיש לנו.

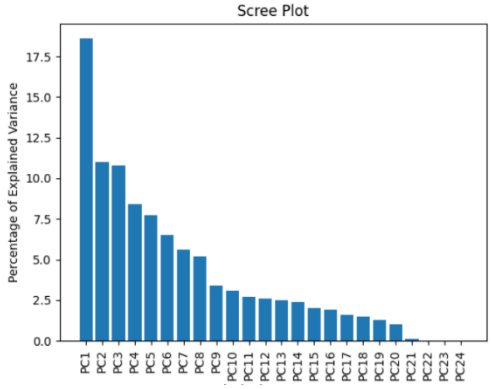
First view



בגרף למעלה ניתן לראות עבור כל משתנה בנפרד איך מתנהגים הלייבלים, אומנם כדי לראות איך כל שלושת המשתנים האלה ביחד מסבירים את המוסבר נרצה להוריד את המדד על ידי שימוש בשיטת PCA שהיא בתורה מקבצת את כלל המשתנים בחלוקת משקלים מסוימת לכל משתנה ומהווה סוג של שכלול לכל המשתנים.בדקנו במקרה של התייחסות לכלל המשתנים איך הגרף של השונות המוסברת של כל PCAים האפשריים, אבל לצורך העניין יותר יעניין אותנו במקרה שבחרנו עבור שלושת המשתנים, כך שניתן לראות שעבור בעיה של שני ממדים של PC1 ו- PC2 נקבל בערך שונות מוסברת מצטברת של 83%

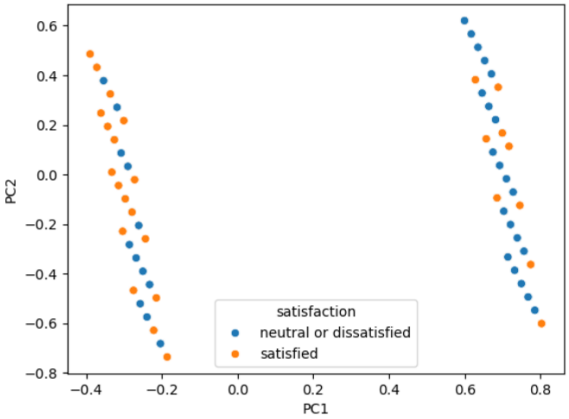
```
[0.54261056 0.2826176 ]
0.8252281601843248
```





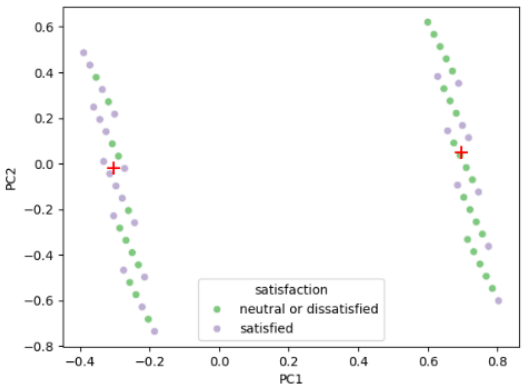
כעט נציג את התוצאות של הסיווג אחרי הורדת ממד.

בעקבות גודל הנתונים הגרף לא יציג את כל הנקודות עקב מגבלות תצוגה.

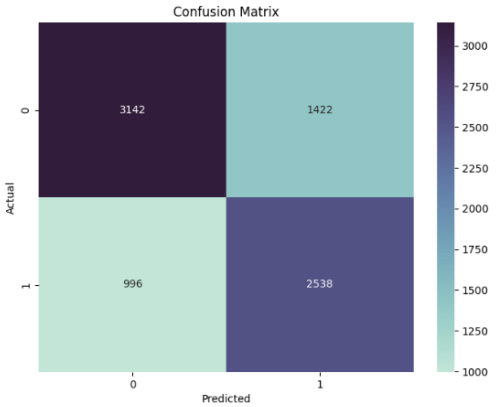


PC1	PC2	satisfaction
0.68672	-0.20285	neutral or diss...
-0.17174	0.15031	neutral or diss...
-0.65254	-0.41184	neutral or diss...
-0.64165	-0.66566	neutral or diss...
0.68672	-0.20285	neutral or diss...
-0.13874	-0.19703	neutral or diss...
-0.19011	0.33285	neutral or diss...
0.74599	-0.56759	neutral or diss...
-0.18295	-0.00999	satisfied
-0.17900	0.07258	neutral or diss...
-0.14944	0.15709	neutral or diss...
-0.15709	-0.12125	satisfied
-0.62659	-0.84336	neutral or diss...
0.69440	-0.03126	neutral or diss...
0.70886	-0.40312	neutral or diss...

הרצנו קודם בהסתמכות על הנתונים שלנו k=2 כך שמדובר בשני ליבילים וקבלנו גרף זה, כך שהאלגוריתם נתן את האשכול הבא שמדובר בשני ליבילים (קבוצה אחת מצד ימין ושניה מצד שמאל), אבל בכל זאת ניתן לראות שיש לא מעט שגיאות בחלוקה שלו וזה בעקבות הפיזור הלא מובהק של התוצאות.



רצינו לדעת עד כמה בפיצול שלו מדויק ולבחון טיב ההתאמה בין האשכולות למחלקות, דבר זה ביצענו על ידי שימוש במטריצת המבוכה שמחושבת על נתוני האימון. ב- 3142 מהמקרים שבהם חזה שמדובר בלקוח לא מרוצה הוא כן חזה נכון 68.84% ובמשלים הוא טעה. ב-2538 מהמקרים שבהם חזה שלקוח מרוצה הוא כן חזה נכון שזה מהווה 71.82% ובמשלים הוא טעה.



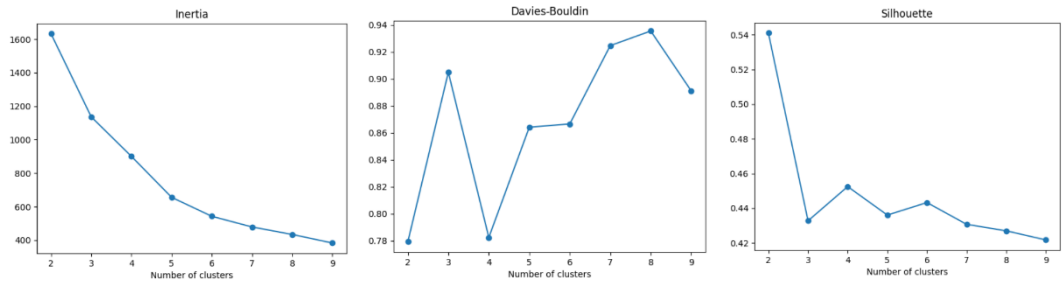
	Pred 0(Accept neutral or dissatisfied)	Pred 1(Accept satisfied)
True 0(neutral or dissatisfied)	TN = 3142 (TNR = 68.84%)	FP = 1422 (FPR = 31.16%)
True 1(satisfied)	FN = 996 (FNR = 28.18%)	TP = 2538 (TPR = 71.82%)

אנו מצפים שניתן לשפר את החלוקה יותר לכן הרצנו עבור Kים שונים בין 2-9. עבור בחירת מספר הקלאסטרים המתאים לנתונים, בחרנו 3 קריטריונים להשוואה: Davies Bouldin, inertia, וSilhouette .

מדד **Silhouette** - ציוני המדד נעים בין 1- ל1 כאשר ציון קרוב יותר ל-1 מצביע על התקבצות טובה יותר עם אשכולות מופרדים היטב. על כן נשאף **לציון גבוה** במדד זה.

מדד **Davies Bouldin** - מדד המשמש להערכת איכות האשכולות, ערכים נמוכים יותר של מדד Davies Bouldin יצביעו על הפרדה טובה יותר וכך על **ציון גבוה** יותר עבור אותו K.

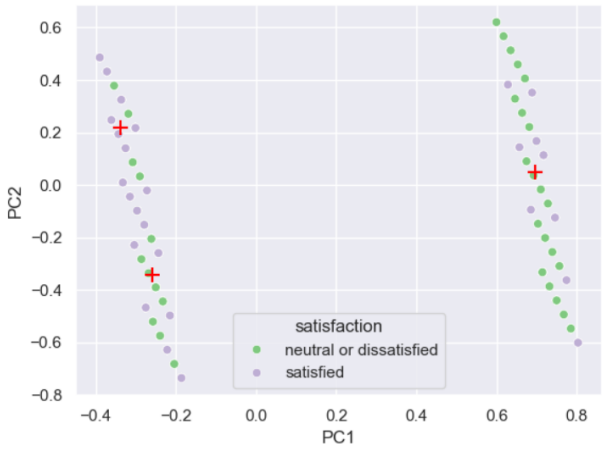
מדד **inertia** - אינרציה משמשת בדרך כלל להערכת האיכות של אלגוריתמי אשכולות, כאשר ערכי אינרציה נמוכים יותר מצביעים על אשכולות הדוקים וספציפיים יותר. על כן נשאף **לציון נמוך** במדד זה.



עבור k=3 מתקבלת הקומבינציה המיטבית ביותר שתניב לנו חלוקה יותר מדויקת עם פחות שגיאות, דבר זה נשמע לנו ממש הגיוני בהסתמכות על המחשבה שהלייבל של ניטרלי או לא מרוצה הוא בעצמו צריך להתפצל לשני ליבילים נפרדים.

אחרי הרצת KMeans עם k=3 התקבל הגרף הבא,

כך שהנקודות הירוקות יש להם שני מרכזים וההנחה היא שבגלל שהנקודות הירקות מצד שמאל רחוקות מקבוצה הנקודות שיש בצד ימין אז ככל הנראה כן מדובר בעוד ליביל נוסף.



אימון מודל נוסף - Random Forest:

מנגנון הסיווג של המודל Random Forest פועל על ידי בניית מספר רב של עצי החלטה. במהלך האימון עץ החלטות ביער נבנה באמצעות תת-קבוצה אקראית של נתוני האימון ותת-קבוצה אקראית של תכונות, מה שמבטיח גיוון בין העצים. בעת סיווג מופע חדש, כל עץ ביער מנבא באופן עצמאי את ה class, וה - class שמקבל את רוב הקולות מבין כל העצים נבחר כתחזית הסופית. תהליך זה של חיזוי ממוצע של מספר עצים מפחית את השונות, מה שמוביל למודלים חזקים ומדויקים יותר בהשוואה לעץ החלטות בודד.

הפרמטרים שבחרנו תחילה הם הדיפולטיבים לקבלת אומדן.
ניתן לראות שקיבלנו שהדיוק לפי f1 הוא 1 בסט האימון, קיבלנו כי קיים
התאמת יתר אך עדין הוא חזה בצורה טובה את סט הוולידציה וקיבל ציון של 0.91 .

```
F1 Score on Training Set: 1.00
F1 Score on Test Set: 0.91
```

לאחר קבלת אומדן של הפרמטרים הדיפולטיבים ניסנו לבדוק איך לשפר את המודל.
ניסינו בעזרת grid search וגם Random Search שזמני הריצה שלו מהירים בהרבה,
עם פרמטרים שונים בכל אחד מהם במגבלת זמן ריצה סביר, קיבלנו:

Random Search: סט אימון 1, סט ולידציה 0.9

```
random forest best
Best Parameters: {'n_estimators': 200, 'min_samples_split': 10, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': 30, 'criterion': 'entropy', 'class_weight': 'balanced_subsample', 'bootstrap': False}
F1 Score on Training Set: 1.00
F1 Score on Test Set: 0.90
```

grid search: סט אימון 0.99, סט ולידציה 0.91

```
random forest best
Best Parameters: {'max_depth': 30, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 200}
F1 Score on Training Set: 0.99
F1 Score on Test Set: 0.91
```

ולכן המודל האופטימלי הוא מה שקיבלנו עם ה grid search.

השוואה בין מודלים:

1. DT או NN לעומת K-Means

עץ החלטות ורשת נוירונאלית הם בהגדרה אלגוריתמי סיווג כלומר, המטרה שלהן היא לסווג כל דוגמה חדשה לאחת הקטגוריות הנתונות, לעומת זאת Means-K היא שיטה לאשכול, כלומר למשימות שבהן המטרה היא לקבץ נתונים דומים לקבוצות (אשכולות) ללא ידע קודם על התוויות של הקבוצות. כיוון שאנחנו רוצים לסווג נתונים חדשים לקטגוריות ידועות נרצה להשתמש ב DT או NN .

2. ביצועי שלושת המודלים

כיוון שמדדנו לכל מודל את מדד הf1 שלו ולפיו גם בחרנו את ההיפר פרמטרים המתאימים, אז זהו יהיה מדד ההשוואה בין המודלי סיווג.

DT - 85.19% על סט האימון ו- 87.19% על סט הוולידציה.

MLP -89.35% על סט האימון ו- 88.56% על סט הוולידציה.

Random forest – grid search: סט אימון 99.0%, סט ולידציה 91.0%.

ניתן לראות כי Random forest נותן את הביצועים הטובים ביותר על סט הוולידציה בפער גדול, אמנם קיים חשש ל overfitting עקב ציון של 99% על סט האימון, אך כיוון שלא הגיע ל100% החלטנו שאין כאן overfitting. לסיכום החלטנו לבחור את המודל Random forest – grid search עם הפרמטרים הנבחרים מסעיף קודם.

המודל הנבחר

1. המודל שבחרנו: Random forest – grid search

ההיפר פרמטרים שנבחרו:

n_estimators = 200

max_depth = 30

min_samples_spli t = 5

min_samples_lea f = 1

max_feature s = sqrt

מודל זה נתן לנו את הציון הגבוה ביותר על סט הוולידציה 91%.

שקלנו לשלב בין כמה מודלים כדי לקבל תוצאות טובות יותר, אבל גנזנו את הרעיון כי ראינו שקיבלנו בסט האימון ציון של 99% כלומר אין הרבה מקום לשיפור המודל בסט האימון ומתקרבים לoverfitting ולכן בחרנו לא לשלב בין מודלים ונשארנו עם Random forest .

לדוגמה הצלחנו לשפר את המודל שנרמלנו את הנתונים בעזרת MinMax במקום StandardScaler בכך שיפרנו את הדיוק על סט הוולידציה באחוז ל 0.92% אך קיבלנו גם שסט האימון הוא 100% ב overfitting ולכן לא נבחר להשתמש ב MinMax לנרמול הנתונים ונישאר עם הנירמול הקודם.

```
F1 Score on Training Set: 1.00
F1 Score on Test Set: 0.92
```

2. ניתוח מטריצת מבוכה:

ישנה הטיה לפי מטריצת המבוכה שמחושבת על נתוני הוולידציה.

ב- 479 מהמקרים שבהם חזה שמדובר **בלקוח לא מרוצה** הוא כן **חזה נכון** 94.48%

ובמשלים הוא טעה. ב-352 מהמקרים שבהם חזה **שלקוח מרוצה** הוא כן **חזה נכון** שזה מהווה 89.57%

ובמשלים הוא טעה.

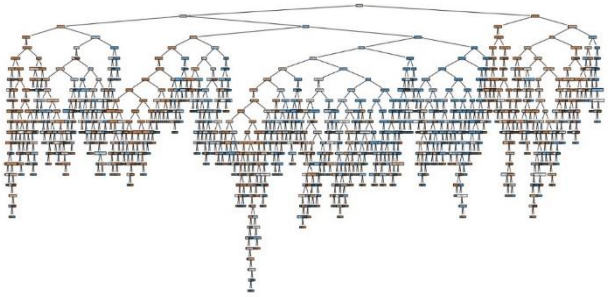
סט הוולידציה הם נתונים שהמודל לא מכיר ולמרות זאת הוא נתן חיזויים טובים של אחוזים גבוהים. דבר שגרם לנו לבחור במודל זה לקראת החיזויים הסופיים.

אפשר לראות חוזקה של המודל בכך שהוא טוב בלחזות לקוח לא מרוצה בצורה נכונה, מאשר לחזות לקוח מרוצה. הדבר טוב לפירמה מכיוון שהיא תוכל לשפר את סך שביעות רצון של כלל הלקוחות בכך שתטרגט את הלקוחות הלא מרוצים.

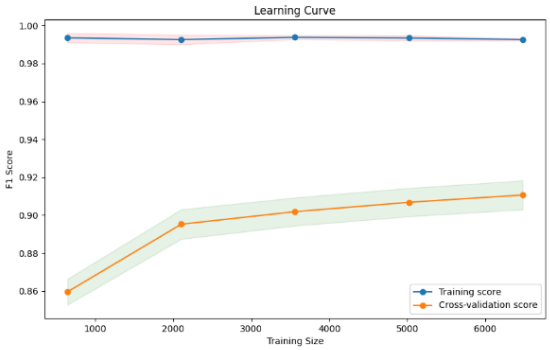
```
                                Pred 0(Accept neutral or dissatisfied) Pred 1(Accept satisfied)
True 0(neutral or dissatisfied)                TN = 479 (TNR = 94.48%)     FP = 28 (FPR = 5.52%)
True 1(satisfied)                               FN = 41 (FNR = 10.43%)  TP = 352 (TPR = 89.57%)
```

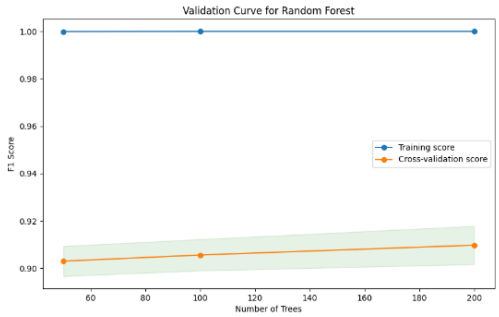
נספחים:

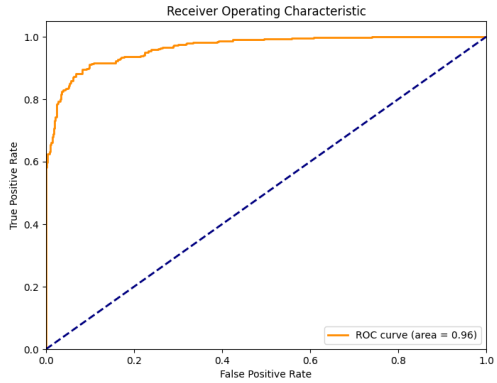
עץ החלטה מלא



גרפים מ Random forest - המודל האופטימלי







מטריצת המבוכה מ Random forest – המודל האופטימלי